Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №5**

**«Исследование алгоритмов сортировки»**

**ПО ДИСЦИПЛИНЕ**

**«Основы алгоритмизации и программирования»**

Выполнил: студент учебной группы

ИСПк-101-51-00

Корюгин Владислав Алексеевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2022

Цель: получить базовые сведения о наиболее известных алгоритмах сортировки, изучить принципы работы с текстовыми файлами.

Задание:

1. Реализовать сортировку данных с помощью алгоритма подсчетом.
2. Реализовать сортировку данных с помощью пирамидального алгоритма.
3. В обоих случаях необходимо предусмотреть возможность изменения компаратора (реализация компаратора в виде передаваемой в подпрограмму функции),
4. Считывание и вывод данных необходимо производить из текстового файла.
5. Для демонстрации работы программных реализаций самостоятельно подготовить варианты входных данных (при этом (объем тестовых файпов должен позволять оценить скорость работы программ).

**Код программы**

uses Crt;

const

N = 100000; // максимальное значение в массиве

type

TArray = array [1..N] of Integer;

procedure Vyvod(var A: TArray; Size: Integer);

begin

writeln('Отсортированный массив:');

for var i := 1 to Size do

write(A[i], ' ');

writeln;

readkey;

end;

procedure Swap(var a, b: Integer);

var

temp: Integer;

begin

temp := a;

a := b;

b := temp;

end;

procedure Heapify(var A: TArray; Size, i: Integer);

var

largest, left, right: Integer;

begin

largest := i;

left := 2 \* i;

right := 2 \* i + 1;

// Поиск наибольшего элемента среди корня, левого и правого потомков

if (left <= Size) and (A[left] > A[largest]) then

largest := left;

if (right <= Size) and (A[right] > A[largest]) then

largest := right;

// Если наибольший элемент не является корнем, то меняем их местами и рекурсивно применяем Heapify к поддереву

if largest <> i then

begin

Swap(A[i], A[largest]);

Heapify(A, Size, largest);

end;

end;

procedure HeapSort(var A: TArray; Size: Integer);

var

i: Integer;

begin

// Построение Max Heap

for i := Size div 2 downto 1 do

Heapify(A, Size, i);

// Извлечение элементов из Max Heap по одному и помещение их в конец массива

for i := Size downto 2 do

begin

Swap(A[1], A[i]);

Heapify(A, i - 1, 1);

end;

Vyvod(A, Size)

end;

procedure CountingSort(var A: TArray; Size: Integer);

var

Count: array [0..N] of Integer;

Sorted: TArray;

i: Integer;

begin

// Инициализация массива Count

for i := 0 to N do

Count[i] := 0;

// Подсчет количества каждого элемента в массиве A

for i := 1 to Size do

Inc(Count[A[i]]);

// Вычисление позиции каждого элемента в отсортированном массиве

for i := 1 to N do

Count[i] := Count[i] + Count[i - 1];

// Перенос элементов из массива A в отсортированный массив Sorted

for i := Size downto 1 do

begin

Sorted[Count[A[i]]] := A[i];

Dec(Count[A[i]]);

end;

// Копирование отсортированного массива в исходный массив A

for i := 1 to Size do

A[i] := Sorted[i];

Vyvod(A, Size)

end;

var

A: TArray;

Size, i: Integer;

InputFile: TextFile;

begin

// Открытие файла для чтения

AssignFile(InputFile, 'input.txt');

Reset(InputFile);

// Чтение размера массива

ReadLn(InputFile, Size);

// Чтение элементов массива

for i := 1 to Size do

ReadLn(InputFile, A[i]);

// Закрытие файла

CloseFile(InputFile);

var m:= 0;

repeat

ClrScr;

writeln('сортировка подсчётом - 1');

writeln('пирамидальная сортировка - 2');

m:=ReadInteger('select program:');

case m of

1: CountingSort(A, Size);

2: HeapSort(A, Size);

end;

until m = 0;

end.

**Результат выполнения программы**

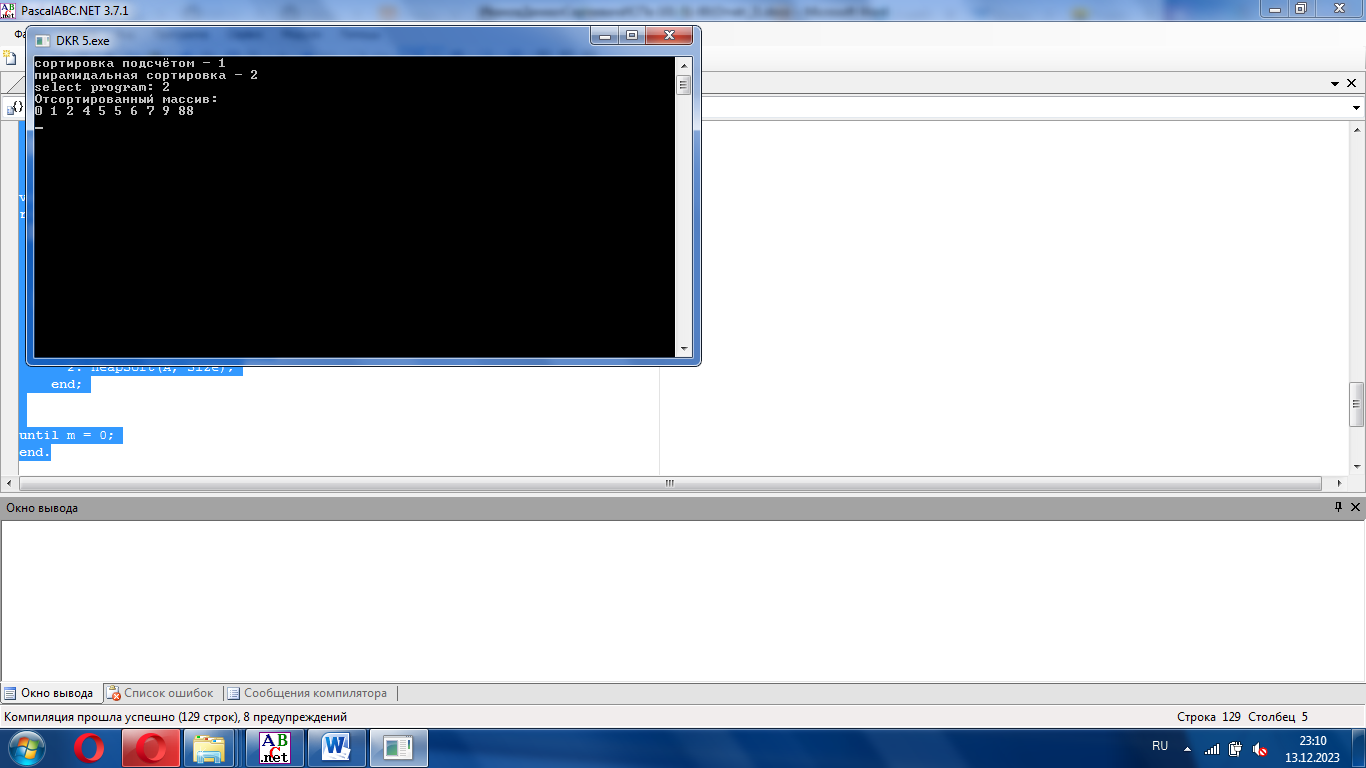


Рисунок 1 – Результат выполнения программы

**Вывод**

В ходе выполнения данной контрольной работы, были получены базовые сведения о наиболее известных алгоритмах сортировки, а также изучены принципы работы с текстовыми файлами. Была реализована сортировка данных с помощью “пирамидального”, а также подсчетной алгоритмов. Считывание входных данных производилось из текстового документа, отсортированные данные записывались в созданный текстовый документ. Работа выполнена в полном объёме.